Análisis paralelo de similitud de imágenes basado en contenido. Incidencia de las comunicaciones según el modelo de arquitectura.

Armando E. De Giusti[1]
Marcelo Naiouf [2]
Laura C. De Giusti[3]

L.I.D.I.[4]**, Facultad de Informática, Universidad Nacional de La Plata**
La Plata, Argentina, 1900
{degiusti, mnaiouf, ldgiusti}@lidi.info.unlp.edu.ar

Resumen

Los algoritmos secuenciales de análisis de similitud de imágenes son de cálculo intensivo, en particular cuando se busca que sean invariantes a traslaciones, rotaciones y cambios de escala. Una solución para el análisis de similitud mencionado se basa en múltiples "firmas", utilizando los coeficientes de la transformada de wavelet. En esta clase de solución se pueden verificar las relaciones teóricas entre las principales componentes del tiempo de procesamiento y el tamaño de la imagen, siendo el tiempo total de procesamiento (Tpt) función exponencial del tamaño de la imagen.

Se han propuesto diferentes soluciones paralelas a esta clase de algoritmos, basadas en arquitecturas multiprocesador lineales y no lineales. En particular los autores han estudiado anteriormente el caso de utilizar una grilla bidimensional con procesadores homogéneos, analizando el speedup teórico, la escalabilidad y la eficiencia, en función del número de procesadores y la complejidad de las imágenes. Asimismo se ha estudiado el caso de búsquedas simples (comparación entre dos imágenes) y múltiples (queries sobre series de imágenes).

En este trabajo se analiza el costo de comunicaciones de la solución paralela, considerando su impacto sobre speedup y eficiencia. Asimismo se plantea la incidencia de las comunicaciones en diferentes modelos de arquitectura, en particular con memoria compartida distribuida.

Finalmente se exponen las limitaciones del modelo utilizado (homogeneidad de los procesadores, costo fijo en tiempo de las comunicaciones, independencia del tamaño del bloque trasmitido) y se señalan las líneas de investigación actuales.

Palabras Claves: Procesamiento de imágenes, Algoritmos paralelos, Similitud, Speedup, Escalabilidad, Eficiencia, Overhead de comunicaciones.

---

[1] Investigador Principal CONICET. Profesor Titular Dedicación Exclusiva. Facultad de Informática. UNLP
[2] Profesor Titular. Facultad de Informática. UNLP
[3] Becaria UNLP. Auxiliar Docente Dedicación SemiExclusiva. Facultad de Informática. UNLP
[4] TE/Fax +(54)(221)422-7707. http://lidi.info.unlp.edu.ar

1. INTRODUCCIÓN

El análisis de similitud de imágenes basado en contenido es una derivación de los problemas de reconocimiento de patrones tradicionales. Se puede descomponer el problema de la siguiente manera:

a Obtener una representación compacta de la imagen y cada "una" de las regiones que contienen objetos que podrían considerarse representativos [1] [2].

b Establecer un modelo de similitud a fin de tener une métrica asociada con la porción del área de dos imágenes que son "similares" [3].

c Reconocer objetos similares a diferente escala, trasladados o rotados en las dos imágenes.

La Figura 1 muestra dos imágenes que debieran reconocerse como “similares” de acuerdo con esta especificación:

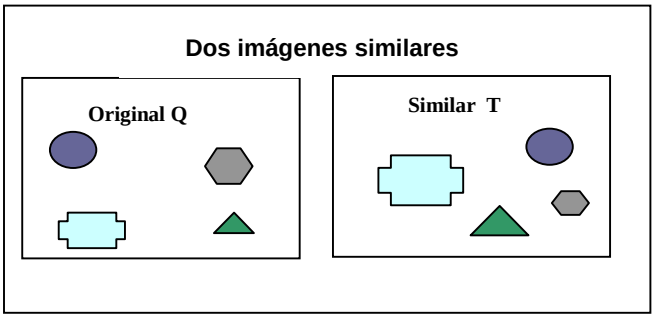

Figura 1

Los enfoques clásicos para análisis de similitud basado en contenido computan algún "rasgo firma" basados en histogramas de color, textura o coeficientes funcionales como la transformada de wavelet [5] [6] [7]. Estas soluciones, basadas en un atributo único de la imagen fallan en los casos de imágenes con componentes escaladas, rotadas o trasladadas. Una alternativa también utilizada es la segmentación de imágenes [8] [9] donde objetos individuales son "extraídos" y comparados. Esto es costoso en tiempo de procesamiento y muy difícil de utilizar con objetos escalados [10][11].

WALRUS [12] [13] presenta la idea de ventanas de diferentes tamaños, donde múltiples ventanas deslizantes se utilizan para descomponer jerárquicamente una imagen completa, de modo de “capturar” los objetos y sus características dentro de cada region. La transformada wavelet resulta robusta con respecto a desplazamientos de la intensidad de colores y también tiene en cuenta la información de textura y forma; el uso de ventanas deslizantes soluciona los problemas de escalado y traslación, por lo cual se pueden obtener resultados eficientes en similitud de imágenes con la representación multi-wavelet [14] [15].

La métrica de similitud de WALRUS considera que dos imágenes Q y T son similares si existe un conjunto de pares similares (Q_i, T_i) tal que el área de matching de las regiones (Q_i, T_i), comparada con el área total de las dos imágenes Q y T está por encima de un límite δ especificado por el usuario. El uso de coeficientes de wavelet sobre ventanas deslizantes permite construir un árbol jerárquico para las imágenes comparadas, en el cual las regiones similares (Q_i, T_i) son nodos donde los coeficientes de wavelet difieren en menos de un límite δ_i especificado [Ec. 1].

$$\delta \leq \frac{Area\left(\bigcup_{i=1}^{k} Q_i\right) + Area\left(\bigcup_{i=1}^{k} T_i\right)}{Total \quad Area\,(Q+T)} \qquad \textbf{Ec. 1}$$

2. COMPONENTES DE TIEMPO EN LA SOLUCION SECUENCIAL

Tal como se analizara en el trabajo anterior de los mismos autores [16], el tiempo total de procesamiento secuencial del análisis de similitud de imágenes mediante multi-wavelet se puede sintetizar en:

Tpt = Tds + Trd + Tli + Tsw + Tjp + Tcc + Tfr

Donde los Tij surgen del esquema de algoritmo secuencial que se ve en la Figura 2:

Dadas dos imágenes I_1 e I_2.

Para cada imagen I_I
  Crear y analizar las estructuras de datos. { Tiempo = Tds }
  Leer los datos de la imagen. {Tiempo = Trd }
  Almacenar cada componente RGB. {Tiempo = Tli }
  Para cada componente RGB
    Computar los coeficientes wavelet p/ c/ vent deslizante.
    Insertarlos en el árbol.
    {Tiempo = Tsw}

Para cada imagen I_I
  Para cada componente RGB
    Computar pixels agrupados { Tiempo = Tjp }
    Calcular los clusters comunes entre las dos imágenes. { Tiempo = Tcc }

Computar el porcentaje de similitud p/c/ componente RGB
Determinar la similitud {Tiempo = Tfr }

Figura 2

Tds, Trd y Tli son tiempos secuenciales los cuales dependen linealmente del tamaño de la imagen, y no tendrán cambios en una solución paralela en un multiprocesador con memoria distribuida. Tfr es un cálculo simple para obtener el nivel de similitud entre dos imágenes utilizando la Ec. 1.

Los componentes más significativos son ***Tsw*** (tiempo de calcular y almacenar los coeficientes de las ventanas deslizantes) y ***Tcc*** (tiempo de calcular los clusters coincidentes). Tal como se demuestra en [16] y [17], considerando imágenes de *n*x*n* y ventanas deslizantes a una distancia de 2, se tiene:

$$Tsw = \sum_{i=2}^{\log_2 n} \left(\left(\frac{n-2^i}{2} + 1 \right)^2 \left(2^{2i} - 4\right) T_c \right) + T_{insert}$$

donde T_c es el tiempo de copiar un pixel.
y T_{insert} es el tiempo para insertar los elementos en un árbol para el caso peor:

$$T_{insert} = \frac{k^2 - k}{2} T_c + kT_{assign} \qquad con\ \ k = \sum_{i=2}^{\log_2 n} \left(\frac{n - 2^i}{2} + 1 \right)$$

$$Tjp = k * \left(4 * 4 \frac{k}{2} T_{compare} + T_{add} \right) + T_{assign}$$

$$Tcc = k * \left(\left(\log_2 k * T_{compare} + T_{assign} \right) + 2 * \left(k + T_{agg} \right) \right)$$

donde,

T_{assign} es el tiempo de asignar un valor.
T_{add} es el tiempo de sumar dos valores.
$T_{compare}$ es el tiempo de comparar 2 matrices.
T_{agg} es el tiempo de insertar un elemento en la lista.

Está claro que Tsw es el componente fundamental en la complejidad del tiempo, y puede ser representado como una función dependiendo del tamaño de la imagen. La Figura 3 muestra la componente de tiempo teórico para Tsw.

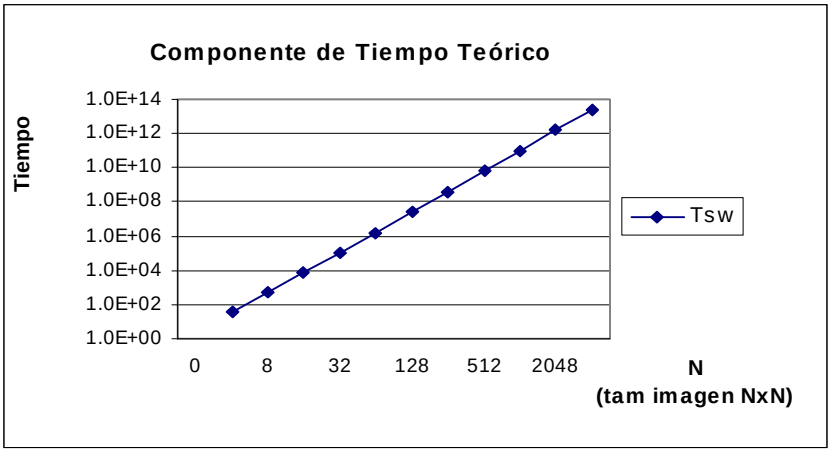

Figura 3

Este análisis de Tsw lleva a pensar en la importancia de paralelizar su tratamiento con el objetivo de obtener un mejor tiempo de respuesta en su cálculo.

3. SOLUCIÓN PARALELA SOBRE UNA GRILLA DE PROCESADORES.

La figura 4 muestra la grilla de procesadores a utilizar

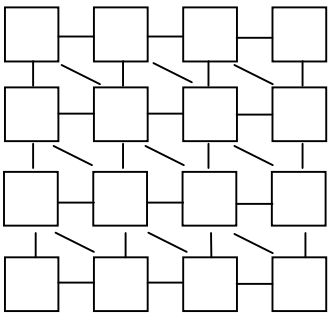

Figura 4

Utilizando una grilla de *P* procesadores, el algoritmo secuencial puede ser paralelizado en $log_2(n)$ pasos. En cada paso se calculan los coeficientes 2^i x 2^i, $\forall i = 1..\log_2(n)$. Este esquema permite una nueva expresión para Tsw:

$$Tswp = \sum_{i=1}^{\log_2 n} \left(\frac{\left(\frac{n-2^i}{2}+1 \right)^2}{min\left\{ P, \left(\frac{n-2^i}{2}+1 \right)^2 \right\}} \left(2^{2i}-4\right) T_c \right) + T_{insert}$$

Cálculo de las ventanas de Wavelet

Dada una imagen I (de *n*x*n*) el algoritmo obtiene la firma de la misma en base a las firmas de sus 4 cuadrantes N_1, N_2, N_3, N_4 (Figura 5a) de la siguiente manera: considerando una nueva descomposición de cada N_i en cuadrantes (Figura 5b), los coeficientes de menor peso correspondientes al 2^{do}, 3^{er} y 4^{to} cuadrante de la ventana I tendrán los valores de los 2^{dos}, 3^{eros}, y 4^{tos} cuadrantes de las subventanas N_1, N_2, N_3, N_4 como se ve en la Figura 5c. Luego, para calcular el cuadrante izquierdo de I, se repite el proceso calculando los coeficientes de ese cuadrante con los coeficientes de los cuadrantes superiores izquierdos de tamaño *n/4*x*n/4* de N_1, N_2, N_3, N_4. El proceso recursivo termina cuando $N_1[1]$, $N_2[1]$, $N_3[1]$, $N_4[1]$ contienen un único valor. En este punto los valores del cuadrante superior de *2*x*2* de I se obtienen realizando el promediado horizontal y vertical de los valores $N_1[1]$, $N_2[1]$, $N_3[1]$, $N_4[1]$.

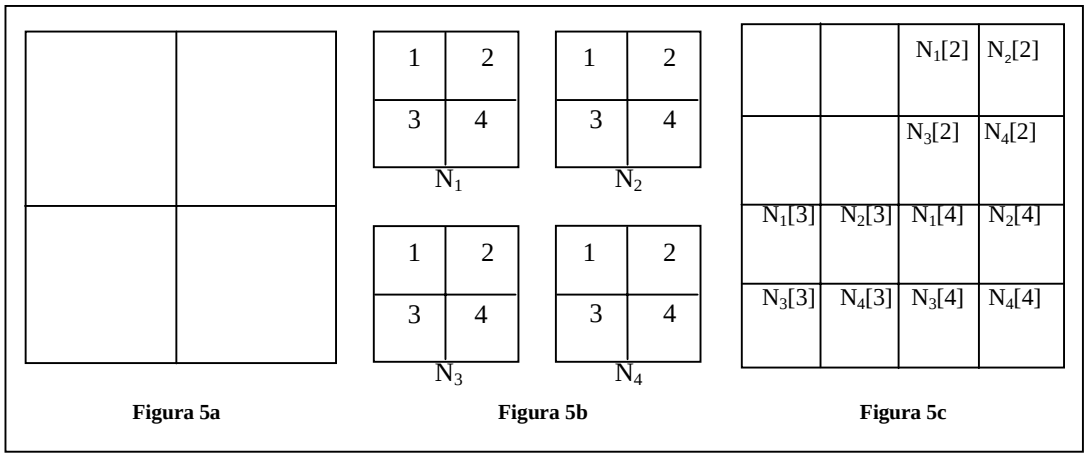

Figura 5a **Figura 5b** **Figura 5c**

Notas a tener en cuenta:

1- La cantidad de ventanas Vi calculadas en cada etapa depende del tamaño de la imagen relacionado con el desplazamiento que existe entre cada ventana. En este caso se utiliza un desplazamiento de 2 pixels tanto en dirección horizontal como vertical.

2- Para calcular las ventanas de un tamaño J deben estar calculadas todas las ventanas de tamaño J/2, salvo las ventanas de tamaño *2*x*2* que se calculan con promedios horizontales y verticales.

Utilizando una grilla de $P = \frac{n}{2} x \frac{n}{2}$ procesadores, la cantidad de procesadores es mayor o igual que la cantidad de ventanas V_i a calcular en la etapa *i*. Esto no ocurre para grillas con menor cantidad de elementos de procesamiento; esto es, si $P < V_i$ en una etapa dada, un mismo procesador debe calcular más de una ventana.

Aunque usar una grilla de $P = \frac{n}{2} x \frac{n}{2}$ procesadores resulta en mayores valores de speedup, una grilla con menor cantidad es una solución más realista. Reducir el número de procesadores afecta el speedup teórico, pero también reduce la comunicación que como se verá posteriormente es un factor crítico en las implementaciones reales.

Las Tablas 1 y 2 muestran los tiempos Tsw paralelo y el speedup para diferentes configuraciones de grilla y tamaños de imagen. Dado que no se ha considerado el overhead de comunicaciones, el resultado es previsible con mejor perfomance para las grillas con mayor número de procesadores [16].

Pixels / Grilla	128x128	256x256	512x512	1024x1024	2048x2048	4096x4096
n/2 x n/2	6,54E4	2,62E5	1,05E6	4,19E6	1,67E7	6.71xE7
n/8 x n/8	1,43E5	5.64xE5	2.24xE6	8.93xE6	3.56xE7	1.42xE8
n/16 x n/16	4,24E5	1.66xE6	6.60xE6	2.38xE7	1.04xE8	3.73xE8
n/64 x n/64	6,04xE6	2,38xE7	7,38xE7	3.73xE8	1,48xE9	5.95xE9

Tabla 1. Tsw paralelo. (sin comunicaciones)

Pixels / Grilla	128x128	256x256	512x512	1024x1024	2048x2048	4096x4096
n/2 x n/2	367	1437	5723	22552	90004	357495
n/8 x n/8	168	666	2676	10587	42338	168341
n/16 x n/16	57	225	908	3965	14393	64188
n/64 x n/64	4	16	82	254	352	614

Tabla 2. Speedup sin comunicaciones (en base a Tsw paralelo)

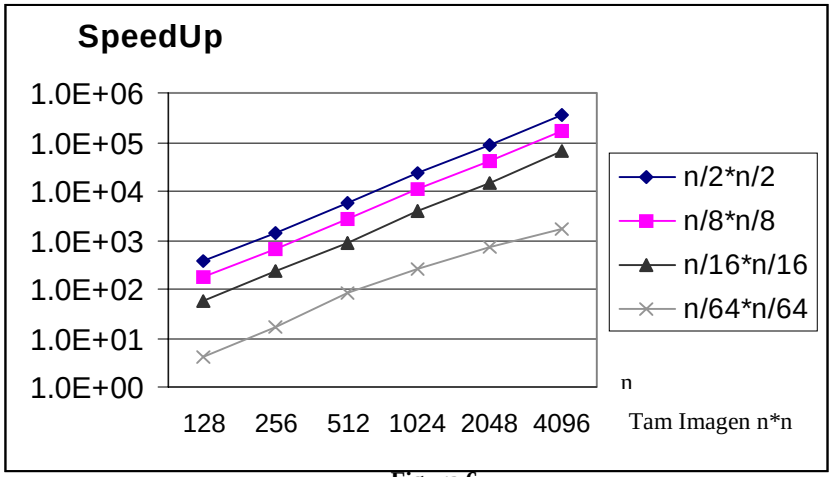

Figura 6

4. INCIDENCIA DE LAS COMUNICACIONES EN LA SOLUCIÓN

4.1. Número de operaciones de comunicación

Tal como se explicó, el proceso de cálculo de las ventanas con los coeficientes de las transformadas wavelet se desarrolla por etapas donde en cada paso se calculan los coeficientes 2^i x 2^i, $\forall$i = 1..log_2(n). Con este esquema, las comunicaciones siguen un esquema similar con la etapa i entregando sus

coeficientes a la etapa i+1. Como muestra la Figura 4, cada ventana de la etapa i+1 requiere de 3 ventanas de la etapa anterior [12] [13]

Simplificando el análisis de las comunicaciones, considerando que el tiempo de startup es fijo en cada comunicación y que las mismas se hacen por paquete, no dependiendo del número de datos de cada paquete (lo cual es real para los tamaños estudiados y los enlaces de comunicaciones utilizados), se puede obtener la Tabla 3 con el número de comunicaciones en función del tamaño de la imagen, con independencia de la arquitectura del sistema de procesamiento. La cantidad de comunicaciones, para una imagen de *n*x*n*, es $\sum_{i=2}^{\log n} 3\left(\frac{n-2^i}{2}+1\right)^2$.

Pixels / Grilla	128x128	256x256	512x512	1024x1024	2048x2048	4096x4096
n/2 x n/2	4.32xE4	2.18xE5	1.06 xE6	5.00xE6	2.31xE7	1.05xE8
n/4 x n/4	3.13xE4	1.70xE5	8.65xE5	4.22 xE6	1.99 xE7	9.24xE7
n/8 x n/8	2.02xE4	1.23xE5	6.73xE5	3.44xE6	1.68 xE7	7.98xE7
n/16 x n/16	1.04xE4	7.92xE4	4.87xE5	2.68xE6	1.37 xE7	6.73xE7
n/32 x n/32	3.27xE3	4.09xE4	3.13xE5	1.94xE6	1.07xE7	5.49xE7
n/64 x n/64	3	1.26xE4	1.61xE5	1.24 xE6	7.74 xE6	4.27xE7

Tabla 3. Número de operaciones de comunicación

Nótese que este número de operaciones de comunicación crece exponencialmente con la complejidad de la imagen y sería el factor predominante de tiempo en cualquier análisis de solución que no admita comunicaciones paralelas.

4.2. Número de operaciones de comunicación sobre la grilla

Al utilizar una grilla de procesadores tal como la de la figura 4 y un esquema de solución como el explicado anteriormente, en cada una de las etapas del procesamiento de los coeficientes de las transformadas wavelet el número total de comunicaciones puede resolverse utilizando los canales físicos que vinculan procesadores adyacentes, minimizando la cantidad de comunicaciones "secuenciales" (naturalmente esto supone procesadores homogéneos con carga de trabajo equilibrada como es el caso).

Suponiendo una grilla de tamaño máximo (*n/2* x *n/2*) y ventanas deslizantes a distancia 2, en la primera etapa habrá 3 comunicaciones desde cada procesador a los 3 procesadores adyacentes que utilizarán los coeficientes en la etapa siguiente; el proceso se repite en la segunda etapa, con la diferencia que la trasmisión debe ser a 3 procesadores a distancia 2. Análogamente en la tercer etapa será a procesadores a distancia 4 en la grilla, y así sucesivamente. En la última etapa se trasmite a una distancia $2^{\log(n-2)}$. En lo que sigue se contabilizará la cantidad de comunicaciones según la camtidad de "saltos" entre procesadores adyacentes en la grilla.

Por otra parte, si la grilla tiene una dimensión menor al tamaño máximo, lógicamente cada procesador resuelve inicialmente un número mayor de coeficientes de la transformada wavelet y hay una menor cantidad de comunicaciones a etapas superiores.

Con estos conceptos, se puede definir la grilla de *p* procesadores como una estructura de *q*x*q* elementos, donde *q* puede valer *n*/2, *n*/4... $n/2^i$ manteniendo las imágenes de *n*x*n* pixels. El número de comunicaciones "secuenciales" que se deben considerar cuando $q = n/2$ es $\sum_{s=1}^{\log n-2} 3*2^s = 3*(2^{\log n-1}-1)$. En general, cuando $q = n/2^i$, para i desde 0 a *log n*, $Ncs = 3*(2^{\log q-1}-1)$.
Si ahora se tabula Ncs (que servirá para corregir los tiempos y speedup paralelos) se tiene la Tabla 4:

Pixels / Grilla	128x128	256x256	512x512	1024x1024	2048x2048	4096x4096
n/2 x n/2	93	189	381	765	1533	3069
n/4 x n/4	45	93	189	381	765	1533
n/8 x n/8	21	45	93	189	381	765
N/16 x n/16	9	21	45	93	189	381
n/32 x n/32	3	9	21	45	93	189
n/64 x n/64	0	3	9	21	45	93

Tabla 4. Ncs = Número de operaciones de comunicación "secuenciales" en la grilla.

5. SPEEDUP DE LA SOLUCION PARALELA CONSIDERANDO LAS COMUNICACIONES

Para estudiar el speedup en la solución con la grilla de procesadores es necesario establecer una relación entre el tiempo de comunicación Tcom (que como se explicó se considera constante) y el tiempo de procesamiento Tpro (básicamente se han considerado por igual asignaciones, comparaciones, sumas y almacenamiento en memoria).

Esta relación depende de la tecnología de los canales físicos que comunican los procesadores de la grilla, e incluso el manejo de las comunicaciones a través de la misma puede alterar la proporcionalidad con la distancia entre procesadores del tiempo requerido para comunicarlos. En este trabajo se han estudiado 6 relaciones diferentes entre Tcom y Tpro, y se fijó el tamaño de la imagen (1024x1024 y 4096x4096 pixels) obteniendo las dos gráficas de speedup que se muestran en las Figuras 7 y 8.

Es particularmente interesante observar que en función de la relación Tcom/Tpro resulta óptimo un número de procesadores diferente para una dada resolución de la imagen.

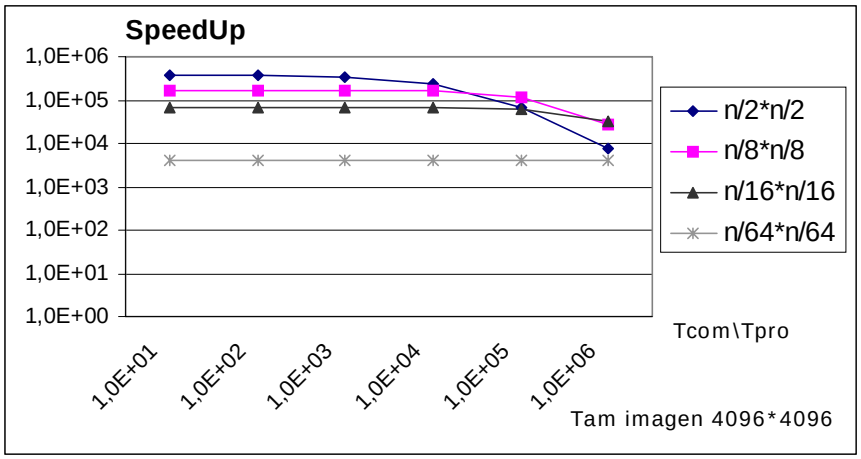

Figura 7

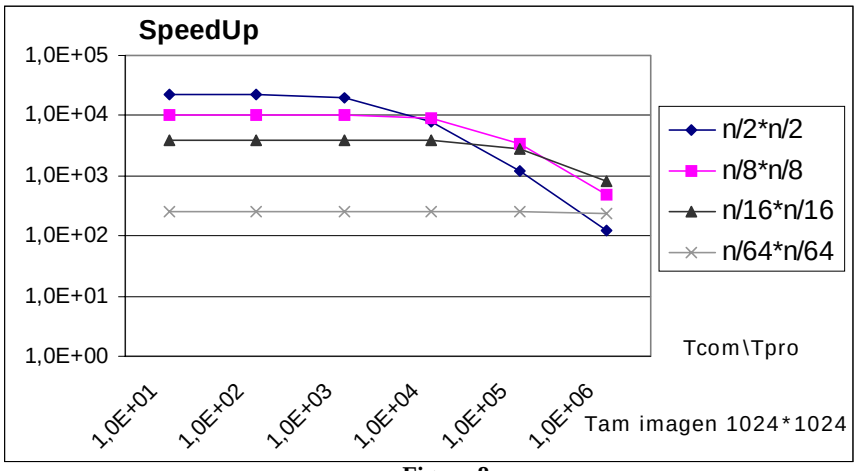

Figura 8

6. EFICIENCIA DE LA SOLUCIÓN PARALELA CON COMUNICACIONES

La eficiencia (igual a speedup/número de procesadores) se relaciona intuitivamente con el menor número posible de procesadores libres a través de los $log_2 n$ ciclos del algoritmo. Como se vio en la Figura 5, si $V[i]$ = número de ventanas cuyos coeficientes deben calcularse en la etapa i ($\forall i = 1..log_2 n$), el número total de procesadores libres es:

$$P_L = \sum_{i=1}^{\log_2 n} (P - V[i]) \qquad \textit{donde } P \textit{ es el número total de procesadores}$$

Los autores habían mostrado en [16] que la eficiencia aumenta con grillas de menor número de procesadores, ya que disminuye sensiblemente el número de elementos de procesamiento ociosos durante el cálculo. Asimismo se vio que la eficiencia promedio mejora notoriamente si se procesa una secuencia de queries de imágenes, ya que los procesadores libres del tratamiento de la imagen j se pueden utilizar para el tratamiento de la imagen j+1.

Este trabajo muestra únicamente el impacto de las comunicaciones sobre la eficiencia alcanzable para las 6 relaciones Tcom/Tpro mencionadas anteriormente (Figuras 9 y 10).

Los resultados son coherentes con el trabajo del 2001 en el que no se tenían en cuenta las comunicaciones: para igual número de procesadores hay una disminución de la máxima eficiencia alcanzable.

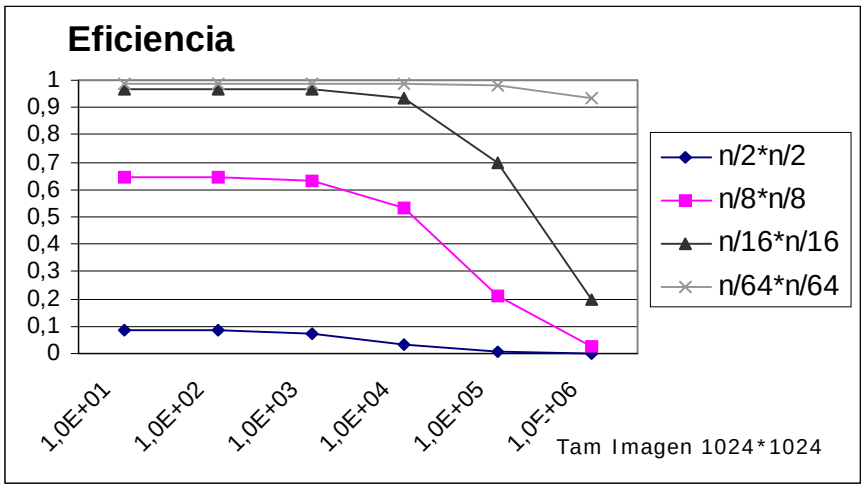

Figura 9

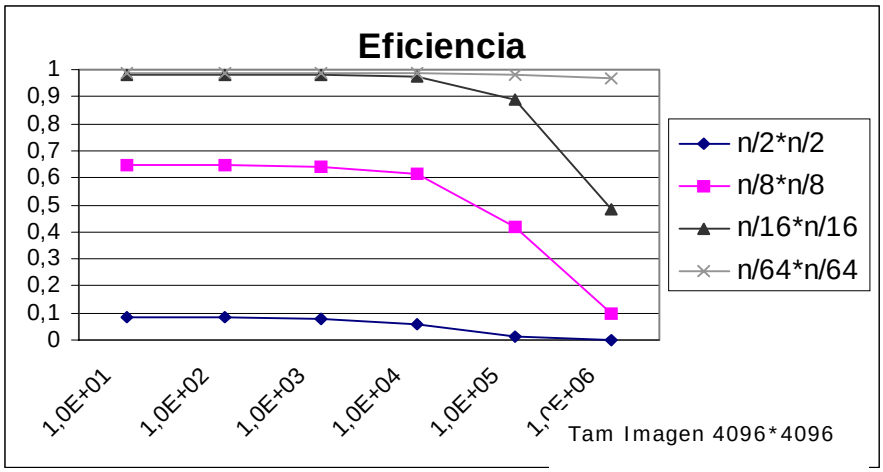

Figura 10

7. RESULTADOS CON OTRAS ARQUITECTURAS

Al considerar las comunicaciones se ve claramente la ventaja de utilizar una grilla de procesadores respecto de una solución tal como un esquema comunicado por bus (en este caso no habría paralelización de las comunicaciones en cada etapa y prácticamente predominaría el costo de comunicaciones).

Por otro lado, si bien el problema pareciera adecuarse a una solución de memoria compartida, un análisis del número de comunicaciones (que se transformarían en accesos exclusivos a memoria compartida) nos muestran que una solución de este tipo resultaría muy ineficiente al aumentar la complejidad de la imagen. En nuestro caso se han realizado algunas mediciones experimentales con la supercomputadora SGI Origin 2000 Clementina [18] ratificando estos conceptos.

Una arquitectura particularmente interesante para la comparación entre dos imágenes es un árbol de procesadores invertido [19] [20]. Si bien no mejoraría el speedup, sí incrementa la eficiencia en razón

del menor número de procesadores ociosos. De todos modos el estudio sobre queries de imágenes (que es el problema de aplicación más significativo que han analizado los autores) vuelve a favorecer la utilización de una grilla regular de *K*x*K* procesadores.

8. LINEAS DE TRABAJO ACTUALES

En este trabajo son muy difíciles las verificaciones experimentales por el número de procesadores requeridos. Por otra parte hay una serie de hipótesis tales como la perfecta homogeneidad de los procesadores y las relaciones de tiempo fijas entre comunicaciones y procesamiento que (si bien lógicas) no necesariamente se mantienen en aplicaciones reales.

Actualmente se están estudiando diferentes problemas de reconocimiento de similitud entre una imagen aislada y bancos de imágenes almacenadas, utilizando el algoritmo básico presentado en este trabajo y variantes de su implementación paralela sobre clusters de PCs y sobre Clementina. Al mismo tiempo se estudia teóricamente el comportamiento de arquitecturas más complejas en función del esquema (y la tecnología) de comunicaciones empleada.

Interesa particularmente la investigación del efecto sobre speedup y eficiencia de la heterogeneidad de los procesadores (incluyendo métricas sobre la misma) y del desbalance de carga de trabajo.

Bibliografía

[1]González R., Woods R., "Digital Image Processing", Addison-Wesley, 1996.

[2]Guibas L.,Rogoff B., Tomasi C. "Fixed window image descriptors for image retrieval", Proceeding Series of Storage and Retrieval for Images and Video Databases III. pp. 352-362-187, 1995.

[3]Jahne Bernd "Digital Image Processing", Springer, 1997.

[4]Zhang T., Ramakrishnan R., Livny M., "BIRCH: An Efficient Data Clustering Method for Very Large Databases", Proceedings of the ACM SIGMOD Conference on Management of Data. pp. 103-114, Montreal, Canada, June 1996.

[5]Niblack W. "The qbic project: Query image by content using color, texture and shape", Storage and Retrieval for Image and Video Databases. pp. 173-187, San Jose 1993.

[6]Jacobs C., Finkelstein A, Salesin D. "Fast multiresolution image querying", Proceedings of SIGGRAPH annual Conference Series. pp. 277-286, 1995.

[7]Castro L., Castro S., "Wavelets y sus Aplicaciones", Proceedings of the I Argentine Congress of Computer Science, Bahia Blanca, Argentina 1995. pp. 195-204.

[8]Hussain Z., "Digital Image Processing" Ellis Horwood, 1991.

[9]Lanzarini,De Giusti, "Class based Clustering" IWISPA 2000, 1st. Int'1 Workshop on Image and Signal Processing and Analysis. IEEE CAS & ASSP Croatia., vol. 21, no. 10, pp.1163-1175, 2000.

[10]Smith J. "Integrated Spatial and Feature Image Systems: Retrieval, Compression and Analysis", PhD Thesis, Graduate School of Arts and Sciences, Columbia University, Feb 1997.

[11] Kalid S., "Introduction to Data Compression" Morgan Kaufmann, 1996.

[12] Natsev A., Rastogi R., Shim K., "WALRUS: A Similarity Retrieval Algorithm for Image Databases", Proceedings of the ACM SIGMOD 1999 Philadelphia. pp. 395-405.

[13] Natsev A., Rastogi R., Shim K., "WALRUS: A Similarity Retrieval Algorithm for Image Databases", Technical report Bell.

[14] De Giusti L., Tarrío D. "Algoritmo de Análisis de Similitud de Imágenes", Graduate Thesine. Universidad de La Plata, Argentina. Dec 2000.

[15] De Giusti A, Naiouf M., De Giusti L "Experimental measures in image similarity analysis", LIDI Technical Report. Universidad de La Plata, Argentina. March 2001.

[16] De Giusti L, Naiouf Marcelo, De Giusti A. "Similarity analisys for image databases. A parallel solution". SCI2001. Florida E.E.U.U. Julio2001. pp 226-231.

[17] De Giusti A, Naiouf Marcelo, De Giusti L. "Paralelización del algoritmo de análisis de similitud de imágenes basado en contenido. Escalabilidad y eficiencia en queries simples y múltiples". Proceedings Cacic 2001. El Calafate, Argentina., pp 745-756.

[18] www.setcip.gov.ar/config_clementina2.htm.

[19] Hwang, K. "Advanced Computer Architecture. Parallelism, Scalability, Programmability". McGraw-Hill, 1993.

[20] Leopold, C. "Parallel and Distributed Computing. A Survey of Models, Paradigms, and Approaches". Wiley Series on Parallel and Distributed Computing. Albert Zomaya Series Editor, 2001.